Group Members:
Pujam Janghel (pjan2245),
Akanksha Patil (apat0532),
Mohammed Saif (moha6885)

ASSIGNMENT: 1 GROUP 6

Tutors: Dr Nguyen Hoang Tran, Iwan Budiman

ABSTRACT

*The purpose of the report is to find the solution that classifies the images into a set of categories as computer is not as compatible as human brain. Thus, the study not only shows the method we used but also how it outperforms other algorithms in different situations. The dataset used here for classification is 30,000 images of size 28 x 28. We classified images into 10 mentioned categories (T-shirt/Top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot) using KNN and Logistic Regression algorithm. However, the best accuracy was achieved by KNN– **84.3%** with K = **5.** This model took **10 seconds** to compute the result.*

TABLE OF CONTENTS

INTRODUCTION

AIM OF STUDY

Human brains can easily differentiate between different clothing items but when it comes to computer systems, it is a difficult task. Our aim is to design an algorithm that can distinguish such clothing items without any human intervention. Hence, we've built a classifier that classifies the images from our dataset into their corresponding categories.

Our dataset consists of images belonging to 10 categories:

1. T-shirt/Top
2. Trouser
3. Pullover
4. Dress
5. Coat
6. Sandal
7. Shirt
8. Sneaker
9. Bag
10. Ankle boot

Classification is a broad ranging research field which includes many decision-theoretic approaches for identifying data. Classification algorithms normally employ two steps – training and testing (2). Classification follows a supervised learning approach, where the computer program learns from input data, and applies this learning to classify new data.

When the dataset is categorized into different classes, various patterns can be observed. These patterns help in revealing insights which would not have been possible prior to classification. Classification problems can either be binary, or multiclass. The selection of classifier depends upon the type of classification required as well as the dataset.

Classifying these images into their corresponding categories also aids further understanding of relationships between them. Identifying these relationships can assist in predicting customer preferences in e-commerce systems as well as online advertising. Another business application of classification can be prediction of a bank loan defaulter, which helps banks in assessing risks before loan disbursement.

METHODS

PRE-PROCESSING TECHNIQUES

Before the data can be used for prediction of values, it needs to be transformed into a format the model can understand. This is where data pre-processing is applied. Data pre-processing not only aids in simplifying the dataset, but also reduces complexity thereby increasing performance and reducing computation time. Another advantage of pre-processing is the increase in prediction accuracy of the model.

The steps that can be followed to pre-process the data are as follows: -

1) Feature scaling:

The process of Feature scaling involves transformation of values of a dataset into a range of (0,1) (Normalisation) without distorting the relation between them. In our case, the data was already rescaled. Rescaling only transforms all the data into a uniform range and does not affect the relationships between the data points. Scaling of the feature vector is necessary to reduce the dominance of features that have a significantly high magnitude of variance compared to others. Feature Scaling helps in speeding up Gradient descent, necessary for Euclidean distance calculation in KNN and critical while performing Principal Component Analysis.

2) Label Encoding:

Our dataset has images which are classified into 10 classes (T-shirt/Top, Trouser, Pullover, Dress, Coat, Sandal, Shirt, Sneaker, Bag, Ankle boot) which can be further referred to as labels. Each of these labels needs to have a unique numeric equivalent- for example, Top- 0, Trouser -1, Pullover -2 etc. Label Encoding assigns unique numbers to each class of the data.

3) Principal Component Analysis:

PCA is a dimensionality reduction technique. For Image classification, PCA transforms the input dataset (images) containing large number of features into a dataset containing smaller number of variables, in order to reduce computation time. In this transformation, the first principal component calculated has the highest variance, and each subsequent component is orthogonal to the preceding component.

CLASSIFIERS

Classifiers are built by implementing classification algorithms to categorize the data into different classes. Classifiers are an instance of supervised learning, which means the machine learns from a training data set which has correctly identified labels. Classifiers are trained on labelled training images, and then used to predict the labels for testing image data. They can work either by analysing individual observations to assign class (For example Logistic Regression) or compare observations to previously identified observation by measuring similarity or distance (K- Nearest Neighbours).

1) K- NEAREST NEIGHBOURS:

INTRODUCTION:

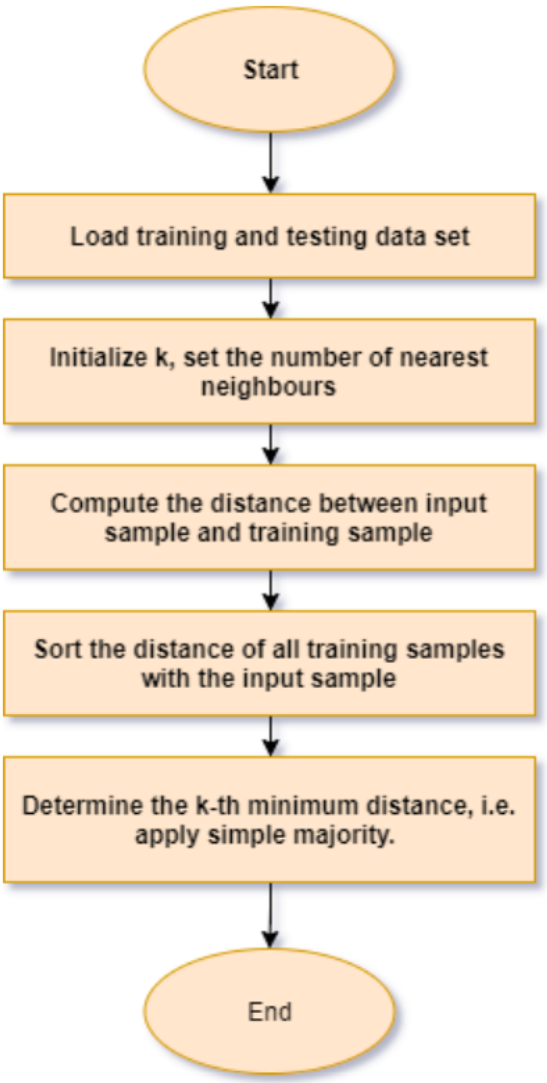
K- Nearest Neighbours algorithm classifies test samples by calculating similarity measure of the testing data with ‘k ’defined number (say 3) of already classified training data points. Computation of similarity measure is a tedious task, especially when dimension of vector increases. In our project we have calculated the similarity measure using Euclidean distance, which is defined as absolute difference between the coordinates of two data points. The classifier then, infers the class of neighbouring data points (which is already known) to predict the class of newly fed data sample.

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

It differs from other classification methods because unlike other algorithms KNN does not formulate any function from the training data instead it simply memorizes the distribution of entire training samples (3). Hence, is also termed as lazy learner.

ALGORITHM:

The algorithm m of K- nearest neighbours is as follows:



2) LOGISTIC REGRESSION:

INTRODUCTION:

Logistic regression is a probabilistic model and works by assigning observations to a discrete set of classes. Logistic regression is a linear regression model and its cost function follows a sigmoidal curve- means its value lies between 0 and 1. Multinomial Regression is called SoftMax regression. Since it works better for predicting discrete values, and the images can only belong to 1 category, SoftMax regression is suitably used for classification problems.

IMPLEMENTATION:

- **SOFTMAX FUNCTION:**

Since our dataset has 10 different classes, K = 10 represented by $C1$, $C2$..., Ck. For each class, we have a parameter vector Wk and the posterior probability will be calculated using the formula below:

$$h_{\mathbf{w}}(\mathbf{x}) = p(y = C_k|\mathbf{x}; \mathbf{w}) = \frac{\exp(\mathbf{w}_k^T \mathbf{x})}{\sum_{k=1}^{K} \exp(\mathbf{w}_k^T \mathbf{x})}$$

- **DECISION BOUNDARY**

Differentiates the probabilities into positive and negative class and is defined by:

$$argmax_k P(C_k|\mathbf{x}; \mathbf{w}) \rightarrow argmax_k \mathbf{w}_k^T \mathbf{x}$$

- **COST/LOSS FUNCTION**

Loss function is used to iteratively calculated at each step using the following formula:

$$\mathbf{J}(\mathbf{w}) = -\left[\sum_{n=1}^{N}\sum_{k=1}^{K} 1_{\{y^n=C_k\}} log(h_{\mathbf{w}}(\mathbf{x}^n))\right] = -\left[\sum_{n=1}^{N}\sum_{k=1}^{K} 1_{\{y^n=C_k\}} log(y^n = C_k|\mathbf{x}^n; \mathbf{w})\right]$$

Where N represents the total number of samples.

- **GRADIENT DESCENT**

Gradient descent is used to optimize performance as it accelerates convergence. It can be calculated through the following model:

$$w_{t+1} \leftarrow w_t - \eta \left(\frac{1}{N} \sum_{n=1}^{N} (h_w(x^n) - y^n) x^n + \lambda w_t \right)$$

Where η represents step size.

EXPERIMENTS AND RESULTS

In order to validate our classifier, we divided the data set in **80 - 20 ratio**: 80% of data set for training our model and the remaining 20% of dataset for validation.

- Total number of records in training set: 24000
  Shape (training set) = 24000 * 784
- Total number of records in validation set: 6000
  Shape (validation set) = 6000 * 784

1) K- NEAREST NEIGHBOURS:

RESULTS:

For k-nearest neighbours' algorithm, we tested our model with different values of nearest neighbours (k). Determining the right value for number of neighbours in KNN is the most important and tricky part. Since there isn't any fixed rule for selecting number of nearest neighbours in KNN it is often considered as very mundane task. The only option remains is to plug in each value and check the accuracy of the model.

Below table shows the accuracy obtained and time taken for our validation set with different number of nearest neighbours.

Number of Nearest Neighbours	Accuracy (%)	Time taken (secs)
1	83.15	10
3	84	11
5	84.3	10
7	83.75	11
9	83.83	12

Key parameters experimentation and **observations** were as follows:

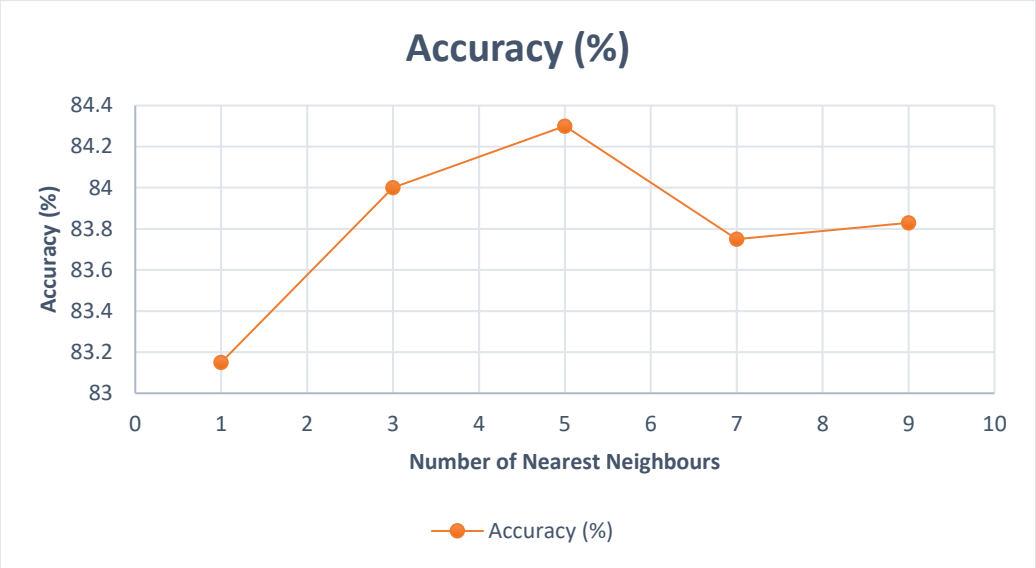
- Number of Neighbours (varied from 1 – 9)
- Total time taken (about 10 - 12 seconds for every value of k)
- Accuracy (82 – 85 % for every value of k)

During our experiments, we found that our KNN classifier had the highest accuracy - 84.3% with K= 5. This model took 10 seconds to compute the result.

EFFECT OF FINE TUNING:

For fine tuning, as stated above, we experimented our model by changing value of number of neighbours and acquired the maximum accuracy for k = 5.

Below graph shows the accuracy obtained for different values of k.



As seen in the graph, the accuracy increased as we changed the value of k from 1 to 5, but it decreased for k = 7 causing underfitting, low variance and high bias. If you choose k as large as the size of your training dataset, variance become equal to zero. Whereas, small value of k leads to overfitting, high variance and low bias.

We tested our classifier on TESTING dataset and got an accuracy of **84.55%** for the first 2000 labels. Please see below output image.

```
1) K-NEAREST NEIGHBOURS
2) LOGISTIC REGRESSION

Please enter your choice and press enter:

1
KNN CLASSIFIER:
The accuracy of the model with KNN is: 84.55 %
Output file containing predicted labels is created by name "predicted_labels.h5" for KNN Classifier
Total time taken by KNN classifier is 4 seconds.
```

2) LOGISTIC REGRESSION:

RESULTS:

Our next highest accuracy was achieved through a logistic regression classifier -> **81.53%** with learning rate set to 0.01. This model took **38 seconds** to compute the result.

We experimented our model with different learning rates. There is no specified algorithm for choosing the most accurate learning rate for our model.

Below table shows accuracy obtained at various learning rates and respective time taken to train the model.

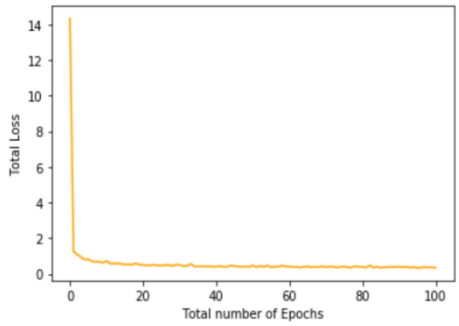
Learning Rate	Accuracy (%)	Time taken (secs)
0.0001	49.73	14
0.001	74.2	33.33
0.01	81.53	38
0.05	80.78	34

Key parameters experimentation and **observations** were as follows:

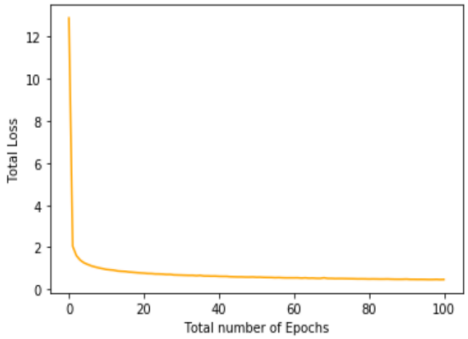
- Learning Rate (0.1, 0.001, 0.01, 0.05)
- Total time taken (between 10 – 40 secs)
- Accuracy (varies from 49% to 82%)

Comparison of loss function for each epoch with respect to different learning rates can be seen below:

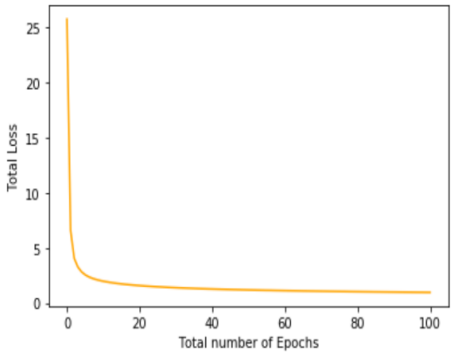
Learning Rate: 0.5



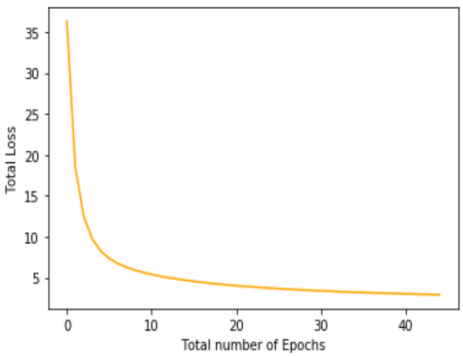
Learning Rate: 0.01



Learning Rate: 0.001



Learning Rate: 0.0001

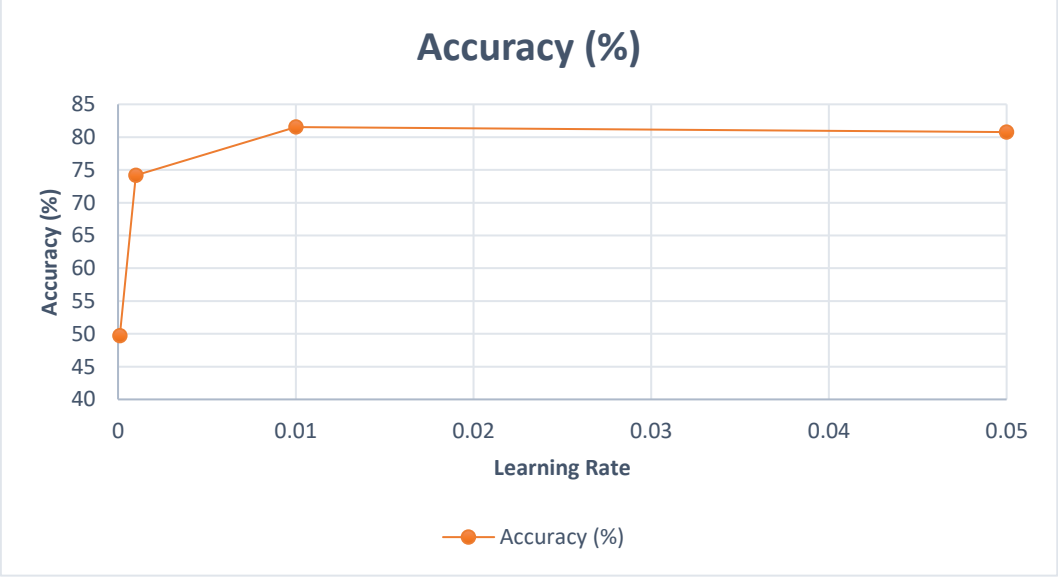


EFFECT OF FINE TUNING:

The learning rate can be tweaked to obtain the highest accuracy as it works as a hyper-parameter which helps in adjusting the weights in our model with respect to loss gradient. Very small learning rate does not result in any significant loss difference thereby causing the process to get stuck, but as the learning rate increases, loss decreases up to a point where it starts to increase. Another constraint faced with the learning rate being too large is that the model will converge earlier than intended producing sub-optimal results. Hence, it is imperative that we choose a suitable learning rate in order to minimize loss and maximize accuracy without negatively affecting performance.

We can fine tune our algorithm by choosing the smallest learning rate in the beginning and then exponentially increasing it with every iteration.

As seen in the below graph, there is a sharp rise in accuracy as learning rate varied from 0.001 to 0.01. Whereas, on further increasing the learning rate, accuracy of the model decreased.



Also, the time required for each learning rate can be seen from the below graph.

DISCUSSION

Our KNN classifier gave the highest accuracy- 84.3%, so it is our best classifier.

- We got the best results with K = 5 and arrived at this number through experimentation. Multinomial Logistic Regression produced an accuracy of 81.53%, which is only slightly lower than KNN. Even though Multinomial Logistic Regression has lower accuracy currently, we feel that with appropriate pre-processing methods, this can be improved.

- KNN classifier was simple and easy to implement. Even though it is simple, it gave us our best accuracy. Another benefit of KNN is that the data point's neighbors provide an explanation for the classification result.

- The very thing that makes KNN so easy to implement can also be called a drawback- no hyperparameters. Due to this 84.3% is its highest achievable accuracy and can't be increased further. Also, there is no fixed way to find the most optimal K. Only way to determine the best value of K is through experimentation, which can be quite tedious. Lastly, since it is a real time algorithm, computation time might suffer for larger datasets, and it may not be the best choice where quicker decisions are required.

- Multinomial Logistic Regression was our second choice for a classifier because it is efficient, gave us good accuracy in acceptable computation time and has parameters that can be optimized. Accuracy and performance can also be improved using pre-processing techniques. It will have better computation time than KNN for large datasets while maintaining accuracy.

- Even though it is efficient, logistic regression is vulnerable to overfitting (5). There are other complex algorithms that can perform better logistic regression (5). Also, since convergence depends upon η (step size) it is imperative to choose an appropriate η. If η is too large, then convergence will be achieved prematurely leading to suboptimal results.

PERSONAL REFLECTION

- Our data is already in normalized form therefore it is used as is in the Classifier. Our classifier is designed to take label encoded data as input for training as well as validation. Since, our data set contains 30,000 images of 28x28 pixels, we initially decided to use Principal Component Analysis to reduce the dimensions of the image. Intuitively, we thought Principal Component Analysis would increase our accuracy, but during our experimentations, we found that using PCA was negatively affecting our accuracy, and therefore we had to forgo its use.

- We first developed our Logistic Regression model and got 81.53% as our best accuracy. We implemented KNN as our second classifier and got a higher accuracy of 83.3%. We also observed that KNN had an average compute time of around 11 seconds across the different K values we tested it with, while Logistic Regression accuracy as well as compute time varied with different learning rates.

CONCLUSION

Even though KNN performed the best in our case, research has shown that its performance suffers for larger datasets (4). Firstly, each neighbour is equally important in the standard KNN. Secondly, KNN is prone to be affected by the imbalanced data problem. Large classes always have a better chance to win (2). Evidence theory KNN can be used to further improve performance (2).

Another approach that can be taken for image classification is by using convolutional neural networks. CNNs have achieved accuracies up to 92.54% (6).

SYSTEM CONFIGURATION

OS Name	Microsoft Windows 10 Home
Version	10.0.17763 Build 17763
OS Manufacturer	Microsoft Corporation
System Manufacturer	Dell Inc.
System Model	Inspiron 5584
System Type	x64 - based PC
Processor	Intel® Core™ i7-8565U CPU @ 1.80GHz, 1992 Mhz, 4 Core(s), 8 Logical Processor(s)
RAM	16 GB DDR4
GPU	NVIDIA GeForce MX130 4GB GDDR5

Appendix

FOLDER STRUCTURE:

- Algorithm (the root folder): .ipynb file containing Python code is placed in this folder

  File Name: SID1490495249_SID2490278701_SID3490323780.ipynb

- Input (a sub-folder under Algorithm): Please copy the test and training datasets into this "Input" folder before running the code.

  File Names: images_testing.h5, images_training.h5, labels_testing.h5, labels_training.h5

  Please make sure that the file names are correct.

- Output (a sub-folder under Algorithm): Prediction file named "predicted_labels.h5" will be saved in this Output folder. The prediction file will contain predicted labels of the test dataset for the first **5000 images.**

CODE STRUCTURE:

The codes contain two classifiers: KNN and LOGISTIC REGRESSION.

Steps to run the code:

1) Restart the kernel and clear output.
2) Run all the blocks
3) You will be asked for a choice of classifier (as we have included the if-else block for classifiers)
4) Enter your choice and press ENTER. (As KNN is our best classifier, please select KNN classifier – CHOICE 1 as it will generate the required output file.)
5) The output will be displayed
    a. KNN: Accuracy, Time taken will be displayed and output file for predicted labels will be generated
    b. Logistic Regression: Accuracy and time take will be displayed. No output file for predicted labels will be generated as our best classifier is KNN.

REFERENCES

1) Zhang Z. Introduction to machine learning: k-nearest neighbors. *Ann Transl Med*. 2016;4(11):218.doi:10.21037/atm.2016.03.37 https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4916348/
2) Lei Wang, Latifur Khan and Bhavani Thuraisingham. An Effective Evidence Theory based K-nearest Neighbor (KNN) classification. Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence 2008.12.09 https://personal.utdallas.edu/~bxt043000/Publications/Conference-Papers/DM/C186_An_Effective_Evidence_Theory_based_K-nearest_Neighbor_KNN_Classification.pdf
3) Stephan Dreiseitl, Lucila Ohno-Machado. Logistic Regression and Artificial Neural Network classification models: a methodology review. Journal of Biomedical Informatics 2002.10.02 https://www.sciencedirect.com/science/article/pii/S1532046403000340?via%3Dihub
4) Machine Learning Basics with the K-Nearest Neighbours Algorithm. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761
5) The Logistic Regression Algorithm https://machinelearning-blog.com/2018/04/23/logistic-regression-101/
6) Shobhit Bhatnagar, Deepanway Ghosal, Maheshkumar H. Kolekar. Classification of fashion article images using convolutional neural networks. 2017 Fourth International Conference on Image Information Processing (ICIIP) 2017.12.21. https://ieeexplore.ieee.org/abstract/document/8313740